Soft Fault Detection and Isolation in Analog Circuits: Some Results and a Comparison Between a Fuzzy Approach and Radial Basis Function Networks

Marcantonio Catelani and Ada Fort, *Member, IEEE*

***Abstract*—This paper provides a comparison between two techniques for soft fault diagnosis in analog electronic circuits. Both techniques are based on the simulation before test approach: a "fault dictionary" is *a priori* generated by collecting signatures of different fault conditions. Classifiers, trained by the examples contained in the fault dictionary, are then configured to classify the measured circuit responses. The suggested classifiers have similar structures. The first is based on a fuzzy system, obtained by processing fault dictionary data for automatic generation of IF–THEN rules, and the second classifier is based on a radial basis function neural network. The two classifiers are used to detect and isolate faults both at the subsystem and component levels. The experimental results point out that both classifiers provide low classification errors in the presence of noise and nonfaulty components tolerance effects. The fuzzy approach provides better results due to an efficient generation method for the IF–THEN rules that allows adding IF parts in the input space regions where ambiguity occurs.**

***Index Terms*—Analog electronic circuit, fault diagnosis, fuzzy classifier, fuzzy logic.**

I. INTRODUCTION

THERE is a growing interest in the development of automatic tools for testing analog circuits. In fact, although large electronic systems are usually implemented by digital techniques, quite often they interface with the external world through analog devices. As an example, all control systems, even when control is implemented by digital techniques, must take inputs from sensors and provide outputs through actuators. Conditioning, multiplexing, and converting analog signals are tasks that most complex systems have to perform.

In digital implementations, well-consolidated techniques for automated test and fault diagnosis are used, which are based on automatic test pattern generators, scan chains, and built-in self-test (BIST). In the last decades, such techniques have become mature and cost effective [1].

Testing of analog circuits and mixed analog–digital systems is surely less understood, yet strongly relevant for applications. In fact, automated fault detection for analog circuits is subject to many problems, such as the presence of noise, the unknown deviation in tolerances of nonfaulty component values, and the location of soft faults [2]. Soft faults are associated with variations of one (or more) circuit component values outside the tolerance range. As such, soft faults do not change the circuit topology, but the circuit operates outside its specifications with an evident loss in performance.

Even if testing of analog circuits is mainly based on the designer's experience and on specifications of a particular circuit's functionality, some methodologies of general validity have been proposed and studied in recent years [3]–[10]. Among these, simulation before test (SBT) automated fault detection methods [5]–[9] seem to have some advantages when the topology of the circuit under test (CUT) is complex. In fact, they allow reducing the test time. In this context, fault conditions are simulated before the test phase with a predefined input stimuli set applied to the circuit inputs. The circuit responses, measured at some selected test points and suitably coded, represent the CUT "signature" for the simulated fault condition; signatures are subsequently collected in a "fault dictionary." Fault detection and isolation are performed by comparing the measured CUT response with all the "signatures" contained in the dictionary.

Due to the continuous nature of the fault mechanism and the presence of noise, a complete "fault dictionary" containing all feasible fault examples cannot be generated. As a consequence, the fault space is sampled, and the diagnosis system is constructed so that a classifier generalizes from the finite set of fault signature examples.

Neural classifiers trained on the fault dictionary examples have been successfully applied to the diagnosis problem [10]–[13], providing satisfactory results.

Recently, fuzzy theory has proved to be a challenging tool in solving pattern recognition problems [14]–[20]; for this reason, here we suggest a novel fuzzy-based approach to the fault detection problem as an alternative to the neural one. The fuzzy system is characterized by an automatic generation of the IF–THEN rules and tuning of the membership functions, based on the examples stored in the fault dictionary. This choice is dictated by the complex relationships that associate the fault conditions (classes) to the modifications of the CUT transfer function or fault signatures (input patterns), strictly dependent on the circuit topology. A fuzzy rule set obtained by the knowledge of a domain expert would not represent a general solution to the circuit diagnosis problem, but will be strongly dependent on the particular circuit under test. In the application considered in this paper, the input space of the fuzzy classifier is, in general, characterized by a high dimension due to the complexity of the fault mechanism. As a consequence, a scatter partition of the input space was preferred to a grid-like

Manuscript received May 4, 2000; revised January 10, 2002.

M. Catelani is with the Department of Electronics and Telecommunications, University of Florence, Firenze, Italy (e-mail: catelani@ing.unifi.it).

A. Fort is with the Department of Information Engineering, University of Siena, Siena, Italy.

Publisher Item Identifier S 0018-9456(02)02920-0.

partition, even if the linguistic interpretation of the rules is less evident.

The performance of the proposed fuzzy approach is compared to the performance provided by a neural classifier based on radial basis functions (RBF) [21]–[24] presented by the authors in [13]. The structure of the paper is as follows. In Section II the classifier and the algorithm providing the fuzzy rules from the fault dictionary are presented. The structure of the RBF classifier is briefly given in Section III. Finally, Section IV introduces results obtained by applying the developed fuzzy classifier to two sample circuits for the diagnosis of faults both at the component and at the subsystem level. Such results are then compared to the ones obtained with the RBF network approach.

II. Fuzzy Approach

In this work, a fuzzy classifier is implemented in order to process circuit input–output measurements. The assumption of accessibility is obviously satisfied; in fact, the CUT input represents the controllable node used to inject stimuli while the output represents the observable node used to measure the CUT response (voltage/current) [2]. Each signature is obtained by injecting predefined stimuli into the input of the circuit under test and by measuring the corresponding responses at the CUT output. Sinusoidal waves with fixed amplitude and different test frequencies have been chosen as stimuli. Since each signature is a collection of N samples of the CUT transfer function, the harmonic analysis has been considered. A number of samples and frequencies are identified by performing a sensitivity analysis of the circuit.

In the following, we consider that a CUT signature $\boldsymbol{x} = \{x^1, x^2, \ldots, x^N\}$ is given, where x^n is the n-th sample of the CUT transfer function magnitude. A first-order Sugeno fuzzy system [25] is considered.

1) The considered fuzzy rule is

$$\begin{aligned}&\text{IF } \left(x^1 \text{ is } MF_j^1\right) \text{ AND } \left(x^2 \text{ is } MF_j^2\right) \ldots\\ &\quad \text{AND } \left(x^N \text{ is } MF_j^N\right) \text{ THEN}\\ &\quad \left(y \text{ is } \boldsymbol{a}_{1j}x^1 + \boldsymbol{a}_{2j}x^2 + \cdots \boldsymbol{a}_{Nj}x^N + \boldsymbol{b}_j\right), \quad j = 1, \ldots, J\end{aligned}$$

where MF_j^n denotes the n-th membership function of the j-th fuzzy rule and $\boldsymbol{a}_{1j}, \boldsymbol{a}_{2j}, \ldots, \boldsymbol{a}_{Nj}$, and $\boldsymbol{b}_j$ are vectors belonging to $\Re^M$. M is the number of considered fault classes, i.e., the number of the fuzzy system outputs.

Sugeno's fuzzy systems are characterized by a crisp output; the first-order Sugeno system has singleton outputs moving in a linear fashion in the output space. Because of the linear dependence of each rule on the input variables, the first-order Sugeno system finds a natural application as a supervisor in the control of nonlinear systems based on multiple linear controllers, where the dynamic behavior of the controlled systems is influenced by the input magnitude. In this application, a first-order Sugeno system allows the output of the classifier to be influenced by the larger fault signature components.

2) The membership functions, MF, are Gaussian functions

$$MF_j^n = \exp\left(-\frac{(c_{n,j} - x^n)^2}{2\sigma_{n,j}^2}\right) \tag{1}$$

where $c_{n,j}$ is the membership function position and $\sigma_{n,j}$ is a scaling factor that defines the membership function width.

3) As it can be seen from part (1), sub-expressions concerning different input variables are combined by fuzzy AND operators (T-norm), realized by the arithmetic product. In this case, the IF part of each fuzzy rule can be described by an n-dimensional Gaussian

$$f(\boldsymbol{x}) = \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{c})^T \boldsymbol{C}^{-1}(\boldsymbol{x} - \boldsymbol{c})\right) \tag{2}$$

where $\boldsymbol{C}^{-1} = \operatorname{diag}\left(1/\sigma_1^2, 1/\sigma_2^2, \ldots, 1/\sigma_N^2\right)$ and $\boldsymbol{c} = (c_1, c_2, \ldots, c_N)$ is the Gaussian center position vector in the input space. As it will be shown later, we assume the scaling factors to be equal, $\sigma_1 = \sigma_2 = \cdots = \sigma_N = \sigma$.

It must be noted that, with the above assumptions, a Sugeno 0th order fuzzy system is equivalent to the RBF network [26].

A critical point in the implementation of a fuzzy classifier is the definition of the membership functions.

A straightforward method is the grid partition of the input space. Nevertheless, this simple method is inefficient especially when the dimension of the input space is high (the number of fuzzy rules is K^N where K is the number of MF concerned with each input variable), or when the input data are not uniformly distributed in the input space but clustered in small regions.

For these reasons, we prefer a scatter partition of the input space based on the fault dictionary data distribution. The partition can be obtained by means of the modified version of the growing cell structure algorithm, proposed by Fritzke in [27], [28]. This algorithm is particularly effective since it defines the fuzzy rules with a supervised scheme by taking into account the classifier output error, but it is characterized by a high computational burden. In order to maintain the algorithm accuracy but reduce its complexity, we suggest a modified version.

The modified algorithm is divided into two steps. First, the IF parts of the fuzzy rules are defined for each class separately in an unsupervised manner by taking into account only geometrical criteria. In this way, the computational burden is reduced approximately by a factor of M (M is the number of fault classes considered for the CUT). In the second step, the IF position is up-dated on the basis of the classifier output error evaluated with the data contained in the fault dictionary. This step, which is highly expensive in terms of computational complexity, is used only to perfect the fuzzy rules in order to reduce the classification error. In this last phase, the weights of the THEN part, $\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_N$, and $\boldsymbol{b}$, are determined in a supervised manner.

In more detail, by starting from a set of training data, the algorithm builds a cell structure in the input space formed by a number of cells connected by edges, which defines both the position and the width of the fuzzy rule IF parts. In the following, the position of a cell is indicated by $\boldsymbol{c}$ and corresponds to the IF position in the input space, while the parameter σ that describes the IF width is given by the average of a subset of edge lengths.

A. Unsupervised Algorithm

We already mentioned that the cell structure is built in two successive steps. A first guess of the structure is obtained by dividing the fault dictionary data into classes and by applying the unsupervised algorithm that will be described in this subsection to each class data separately.

The unsupervised algorithm starts from a simple structure made up of $N+1$ (N dimension of the input space) cells all connected by $(N+1)N/2$ edges (hyper-tetrahedron or simplex). The starting structure for $N = 2$ is a triangle. The cell structure A grows by means of an iterative procedure to represent the fault dictionary data distribution. A local counter variable τ is initialized at zero for each cell of the growing structure. The fault dictionary data are processed randomly, and the entire set is processed several times.

The cell structure A is adapted to the data distribution by selecting an input data $\boldsymbol{x}$, finding the best matching cell $\boldsymbol{c}_b$, i.e., the cell characterized by the minimum geometrical distance from the considered data and moving this unit, $\boldsymbol{c}_b$, according to the following equation

$$\Delta \boldsymbol{c}_b = \varepsilon_b(\boldsymbol{x} - \boldsymbol{c}_b) \tag{3}$$

where ε_b is an adaptation parameter.

The local counter of the best matching unit is then incremented. Also, the topological neighbors, $\boldsymbol{c}_t$, of cell $\boldsymbol{c}_b$, are moved, since the topological neighbors are defined as those cells connected by an edge to cell $\boldsymbol{c}_b$.

New cells are added after a fixed adaptation step number in the input space regions characterized by a high density of input vectors (fault dictionary examples). In particular, a new cell is added between the cell with the higher value of the associated local counter τ, and its furthest topological neighbor. The newly inserted cell is connected by new edges to the existing ones, so that the structure maintains its topological structure.

Ideally, at the end of the adaptation process, all the local counter variables should have approximately the same value. Hence, the algorithm stops when the maximum difference in the local counter variable relative to the number of processed data falls below a fixed threshold.

B. Supervised Algorithm

The unsupervised adaptation algorithm described above iterates M times. At the end of this phase, M different cell structures are obtained, which are merged to form a unique structure with the same topological properties. The resulting net is further adapted by means of a supervised version of the same algorithm.

In the following, we assume that the data contained in the fault dictionary are couples of $\left(\boldsymbol{x}_i \in \Re^N,\, \zeta_i \in \Re^M\right)$ type; $\boldsymbol{x}_i$ is the input data, i.e., the fault signature, while ζ_i is the classifier desired output. In particular, ζ_i is the M-dimensional vector with a "1" in the position corresponding to the fault class for $\boldsymbol{x}_i$ and zero elsewhere.

In this contest, the fuzzy system has N inputs and M outputs. The classification criterion is: *input belongs to the class that corresponds to the largest fuzzy system output*.

In the supervised scheme, the fuzzy system is trained to perform an optimal classification of the data contained in the fault dictionary, by iteratively adapting both the IF part (center and width) and the THEN part. Note that the parameters of the j-th THEN parts $\boldsymbol{a}_{1j}, \boldsymbol{a}_{2j}, \ldots, \boldsymbol{a}_{Nj}$, and $\boldsymbol{b}_j$ are M-dimensional vectors, for instance $\boldsymbol{a}_{1j} = (\boldsymbol{a}_{1j}^1, \boldsymbol{a}_{1j}^2, \ldots, \boldsymbol{a}_{1j}^M)$, relating the j-th IF part to each of the M classifier outputs.

The classifier parameters are evaluated at the first step. Given the first guess of the growing cell structure obtained by the unsupervised algorithm.

- The center of the j-th fuzzy IF corresponds to the cell $\boldsymbol{c}_j$ position.
- The width of the j-th IF (Gaussian) is given by the average of the length of the two shortest edges connecting cell $\boldsymbol{c}_j$ with the others.
- The value of each IF part is evaluated as

$$f_j(\boldsymbol{x}_i) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{c}_j\|^2}{\sigma_j^2}\right). \tag{4}$$

- The m-th output o_m is evaluated as

$$o_m = \sum_{c_j \in A} \left(a_{1j}^m x_i^1 + a_{2j}^m x_i^2 + \cdots + a_{Nj}^m x_i^N + b_j^m\right) f_j \qquad \forall\, m \in \{1, \ldots, M\}. \tag{5}$$

Note that, at the first step, the weights of the j-th THEN are preset by selecting values that minimize the output error with a least mean square error function.

At this point the algorithm described in Section II-A is applied with a different selection of the local counter variable. It is updated by the classification error

$$\Delta\tau_s = \|\zeta - \boldsymbol{o}\|^2 \tag{6}$$

where $\boldsymbol{o} = (o_1, o_2, \ldots, o_M)$.

The weights of the THEN parts are updated at each iteration as

$$\begin{aligned} \Delta a_{nj}^m &= \eta(\zeta_m - o_m) f_j x_i^n \\ \Delta b_j^m &= \eta(\zeta_m - o_m) f_j. \end{aligned} \tag{7}$$

When a new cell is inserted, its associated THEN weight vector must be generated.

In this phase, the algorithm stops when the variation of the output mean error is below a fixed threshold.

With the proposed procedure based on unsupervised and supervised training, it is possible to ensure a reduced computational burden. In fact, the first unsupervised phase is performed on data belonging to each class separately (with a computational complexity that is approximately reduced by a factor $1/M$ with respect to the complexity of the same algorithm when applied to the whole data set). Moreover, the first phase already provides a good approximation of the final structure.

The second phase, characterized by a higher computational burden being applied to the complete training data set, consists, in general, of a reduced number of adaptation steps and allows inserting cells in the input space regions where the classification task is more complex. Please note that no cells are added in the region of the input space where no mis-classifications occur. The number of IF parts grows in those regions of the input space where classification is more difficult. Finally, it must be taken into account that in this phase the IF and the THEN parts are simultaneously changed to minimize the output error.

III. Neural Approach Based on RBF Networks

Results provided by the fuzzy approach were compared with those obtained by the authors in a previous work [13] where a radial basis function neural (RBF) classifier was suggested.

RBFs are layered neural networks with the hidden layer characterized by Gaussian functions (or radial basis functions) [21]–[24], [29], [30]. Given an input $\boldsymbol{x}$, activation $\boldsymbol{a(x)}$ for the hidden units is defined as

$$a(\boldsymbol{x}) = \exp\left(-\frac{(\boldsymbol{c}-\boldsymbol{x})^2}{2\sigma^2}\right) \tag{8}$$

where $\boldsymbol{c}$ is the position in the input space of the radial basis function, and σ is the corresponding scaling factor characterizing the area of the activation region.

The output layer is composed of M linear adding units, linked to the hidden layer by weighted connections.

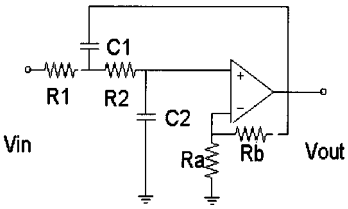
The analogy of the RBF structure with the Sugeno of 0th order fuzzy classifier is apparent [26]. We configure the classifier so that M coincides with the number of fault classes to be detected (each output neuron corresponds to a class of fault). An input belongs to the fault class characterized by the largest value of the output neuron.

The training set is formed in the same way used for the fuzzy approach. The target output vectors have a single "1" in the position corresponding to the correct class and "0s" elsewhere.

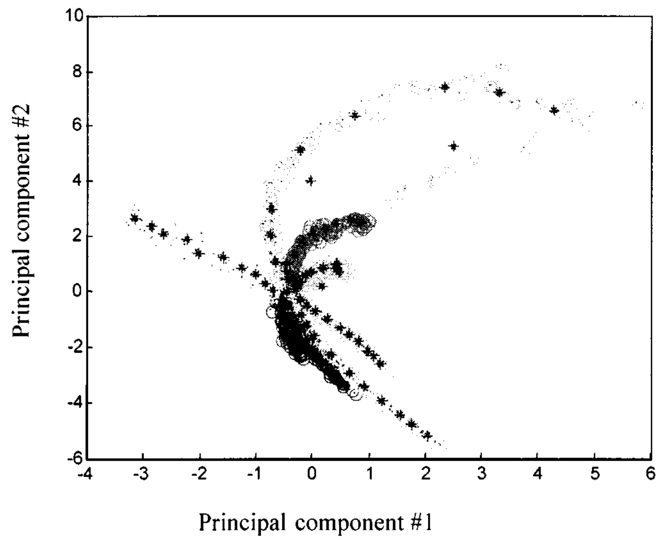
Training is very fast. In fact, an unsupervised technique is first used, which consists of placing the hidden units layer centers on the centroids of training input clusters. A clustering algorithm is applied to input vectors of the training set to determine cluster centroids. In [13], the fuzzy C-Mean clustering algorithm was used to cluster separately data belonging to the different fault classes. In the second step, the width of the Gaussian functions is evaluated (the value depends on the minimum distance among cluster centroids). The weights of the output layer are trained by a supervised process. A least square regression relates the desired network outputs to the hidden node activations; each training example is passed through the first layer and the hidden nodes to produce a corresponding hidden node activation vector. Weights that minimize the squared norm of the difference between the desired outputs and the net outputs are found.

IV. Experimental Section

In this section, we compare the performance of the two classification approaches described in the previous sections. Results have been obtained by taking into account both zero mean white additive noise superimposed to the circuit output response and the component tolerance effects. To this aim, we assume that each nonfaulty component of the circuit under test is a random



(a)



(b)

Fig. 1. (a) Circuit under test (CUT), low-pass filter; (b) "o" circuit signatures where different gray shades denote different fault classes; "*" denotes IF part positions.

variable uniformly distributed in the tolerance range. In order to construct the fault dictionary and compensate for the component tolerance effects, multiple signatures are generated for each potential fault of the CUT. The signatures are obtained by injecting in the controllable node (input) a set of stimuli consisting of sine waves with constant amplitude and different test frequencies selected from the frequency range of the CUT. For each stimulus, the corresponding voltage amplitude is measured at the output test point, and, hence, N samples of the frequency transfer function magnitude are obtained.

In this work, two major categories of data pre-processing were applied to circuit signatures contained in the fault dictionary before using them to train the classifiers: normalization and compression. Normalization, which is commonly employed in conjunction with classifiers, prevents the subsequent pattern recognition algorithms from being biased by signature components with intrinsic larger response magnitudes. Here, data compression is achieved by means of the principal component analysis (PCA) [30].

In order to test the proposed technique, the low-pass filter of Fig. 1(a) was considered. The nominal values and tolerance for each component of the filter are summarized in Table I. For this CUT, single faults at the component level have been taken into account while a sensitivity analysis led to the selection of eight test frequencies. R_1, R_2, C_1, and C_2 were considered to be potentially faulty elements. Faults affecting the resistors R_a and R_b form an ambiguity class and can be grouped in a single class

TABLE I
COMPONENT VALUES AND TOLERANCE RANGES FOR THE COMPONENT OF THE CIRCUIT SHOWN IN FIG. 1(A)

COMPONENT	Nominal value	Tolerance
C_1	50 nF	± 5%
C_2	50 nF	± 5%
R_1	100 Ω	± 10%
R_2	100 Ω	± 10%
R_A	5 kΩ	± 1%
R_B	2 kΩ	± 1%

called "gain fault." The gain fault can be detected by a simple dc measurement, and it is not included in the results reported hereafter. To obtain more accurate results each fault of a component is represented by two different fault classes: a class for component values larger than the nominal one and the other for component values smaller than the nominal. So, eight fault classes plus one class for the operating fault-free condition are obtained in this case. To form the fault dictionary, the faulty component values have been extracted from the uniform distribution defined in the intervals $[0.1Xn;\ (1-t)Xn]$ and $[(1+t)Xn;\ Xn]$, where t is the tolerance range and Xn the nominal value of the circuit element. The input dimension was reduced by PCA from eight to three. This reduction allows the designer to consider the grid partition method (otherwise, it could not have been applied for computational complexity reasons). For the filter of Fig. 1(a), a white additive noise is superimposed to the circuit output and a SNR of 30 dB was considered.

In Fig. 1(b), the projections of the IF part centers, c_j, obtained with the growing cell structure algorithm are shown together with the projections of the vectors contained in the fault dictionary. The projection plane shows the two first principal components of the fault dictionary data. It can be seen that where data of different classes are closer and the situation is more confused, the growing structure has a higher density of cells corresponding to a large number of fuzzy rules. In Fig. 2, the plot of classification error versus the number of fuzzy rules is given. The unsupervised phase of the training algorithm was stopped when the growing cell structure reached a given complexity. Then, the supervised algorithm was used to reduce the error and stopped when the error gradient reached a predefined threshold. Table II reports the results concerning the filter in Fig. 1(a), obtained with 900 training vectors and 1800 test vectors. It can be seen that results obtained with the three methods, fuzzy with grid partitioning, FGP, fuzzy with scatter partition (growing cell structure), FSP, and RBF networks are very similar, but the second method provides better results. It is important to observe that the complexity of the classifiers is similar. In fact, the RBF has 63 hidden nodes, and the scatter partition fuzzy classifier has 74 fuzzy rules.

The same fault diagnosis technique was then applied to a more complex circuit (see Fig. 3). In this case, we considered faults at the subsystem level. The circuit is composed of four

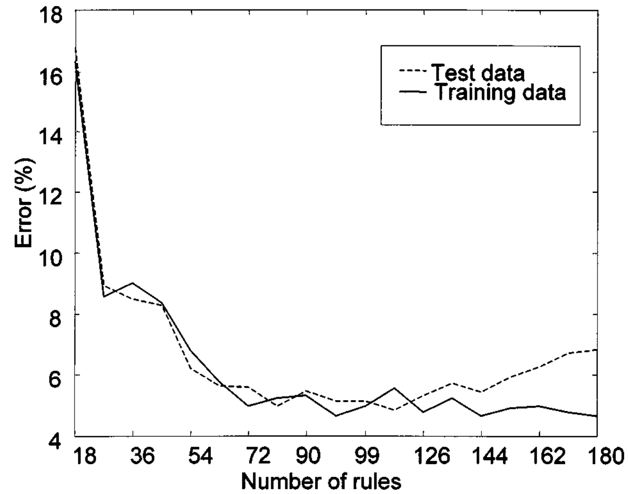


Fig. 2. Classification error as a function of number of fuzzy rules (9 fault classes).

TABLE II
CLASSIFICATION ERRORS OBTAINED WITH FUZZY CLASSIFIER AND GRID PARTITION OF THE INPUT SPACE (FGP); RADIAL BASIS FUNCTION CLASSIFIER (RBF); FUZZY CLASSIFIER WITH SCATTER PARTITION OF THE INPUT SPACE OBTAINED BY GROWING CELL STRUCTURE (FSP)

FAULT CLASSES	FGP ERROR (%)	RBF ERROR (%)	FSP ERROR (%)
No fault	0.08	0.25	0.52
Faulty C_1	4.33	1.89	1.66
Faulty C_2	2.21	1.16	1.32
Faulty R_1	1.94	1.01	0.82
Faulty R_2	2.95	1.79	0.99
TOTAL ERROR (%)	11.50	6.10	5.31

second-order filters plus an adder. For this circuit, five fault classes are considered (with reference to Table III: χ_2: high-pass filter 1 faulty; χ_3: low-pass filter 1 faulty; χ_4: high-pass filter 2 faulty; χ_5: low-pass filter 2 faulty; χ_6: adder faulty), plus a class for the operating circuit (χ_1). Faults are defined as deviations of the circuit frequency response from the nominal one. In particular, there is a fault when one of the subsystems of the circuit does not work properly, i.e., when the amplitude of the frequency response of the filter, evaluated at the nominal cut-off frequency, differs more than 20% from the nominal value. The fault dictionary has been generated by considering single faults for the circuit components responsible for the subsystem fault. In Table III, the component values used for the circuit are listed together with the tolerance range of each component.

The circuit signature is constructed with the same procedure designed for the circuit in Fig. 1(a). Here, 16 frequency samples

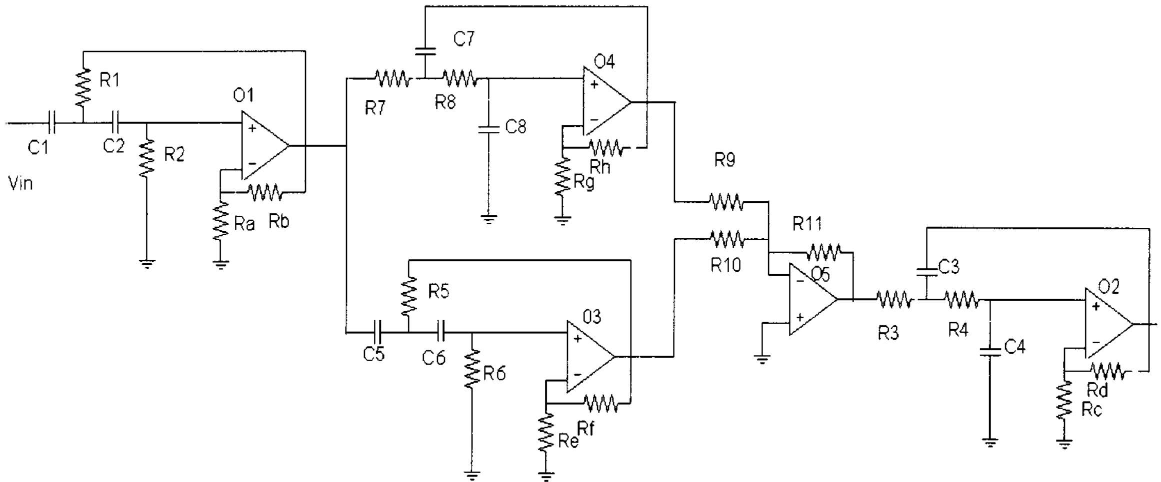


Fig. 3. Circuit under test.

TABLE III
NOMINAL VALUE AND TOLERANCE RANGES FOR THE COMPONENTS OF CIRCUIT IN FIG. 3

COMPONENT	NOMINAL VALUE	TOLERANCE	SUB-SYSTEM
R_1	**320 kΩ**	**10%**	**High-Pass Filter 1**
R_2	**320 kΩ**	**10%**	**F_t=10 Hz**
C_1	**50 nF**	**5%**	
C_2	**50 nF**	**5%**	
A_{v1}	**1.75**	**1%**	
R_3	**32 Ω**	**10%**	**Low-Pass Filter 1**
R_4	**32 Ω**	**10%**	**F_t=100 kHz**
C_3	**50 nF**	**5%**	
C_4	**50 nF**	**5%**	
A_{v2}	**1.75**	**1%**	
R_5	**320 Ω**	**10%**	**High-Pass Filter 2**
R_6	**320 Ω**	**10%**	**F_t=10 kHz**
C_5	**50 nF**	**5%**	
C_6	**50 nF**	**5%**	
A_{v3}	**1.75**	**1%**	
R_7	**32 kΩ**	**10%**	**Low-Pass Filter 2**
R_8	**32 kΩ**	**10%**	**F_t=100 Hz**
C_7	**50 nF**	**5%**	
C_8	**50 nF**	**5%**	
A_{v4}	**1.75**	**1%**	
R_9	**1 kΩ**	**1%**	**Adder**
R_{10}	**1 kΩ**	**1%**	
R_{11}	**1 kΩ**	**1%**	

were taken into account; the fault dictionary is composed of 3000 fault signatures with a 30 dB SNR.

Also in this case, data were preprocessed with PCA to reduce the dimension of the input space from 16 to eight dimensions.

Results, obtained with 30 000 test vectors, are shown in Table IV. It can be seen that for this circuit the error rate is about 8%. Also in this case, a comparison among different methods is shown, which confirms results already found for the CUT in Fig. 3. As expected, the performance of the different methods is very similar, as is the complexity of the classifiers (114 hidden nodes of the RBF versus 122 IF parts of the fuzzy systems). The resulting classification errors are near 8% for all the compared methods.

As already highlighted, the RBF classifier is equivalent to a 0th order Sugeno fuzzy system. It can be seen that the growing cell structure algorithm is a more efficient training algorithm. The slight enhancement in classification is probably due to the supervised refinement of the IF parts (equivalent to the hidden units of the RBF), obtained in the final phase of the proposed training algorithm. The best results obtained with the 1st order Sugeno (7.5% for FS1 versus 8.5% for RBF), are justified by the coupling of the more efficient training (growing cell structure) with a larger number of degrees of freedom (parameters of the THEN parts).

V. CONCLUSION

An automatic diagnosis technique based on a fuzzy classifier was presented and applied to fault location in analog electronic circuits, considering single faults both at the component and subsystem level.

The fuzzy rules are defined by processing a data set contained in the fault dictionary, applying an efficient algorithm based on a growing cell structure. The algorithm is divided in two phases so that the computational burden is reduced, but the advantages of a supervised scheme are preserved. It is shown that, under the assumed hypotheses, the structure of the fuzzy classifier is analogous to that of a radial basis function network. Hence, the performances of the two methods are similar. Results obtained by applying a first-order Sugeno fuzzy system are justified by the efficiency of the classifier training method and the large number of degrees of freedom.

TABLE IV
CLASSIFICATION ERRORS OBTAINED WITH FUZZY CLASSIFIERS AND RADIAL BASIS FUNCTION CLASSIFIER (RBFN)

FAULT CLASSES	RBF ERROR (%)	FUZZY SUGENO 0 ERROR (%)	FUZZY SUGENO 1 ERROR(%)
χ_1	0.52	0.74	0.74
χ_2	0.52	1.35	1.32
χ_3	1.04	1.05	0.99
χ_4	1.73	1.35	1.13
χ_5	1.74	1.60	1.53
χ_6	2.95	1.77	1.73
TOTAL ERROR (%)	8.5	7.89	7.44

REFERENCES

[1] L. S. Milor, "A tutorial introduction to research on analog and mixed-signal circuit testing," *IEEE Trans. Circuits Syst. II*, vol. 45, pp. 1389–1407, Oct. 1998.

[2] J. L. Huertas, "Test and design for testability of analog and mixed-signal integrated circuits: Theoretical basis and pragmatical approaches," in *ECCTD '93 Circuit Theory Design '93: Selected Topics Circuits Syst.*, ch. 2, Davos, 1993, pp. 75–151.

[3] J. A. Starzy and J. W. Bandler, "Multiport approach to multiple fault location in analog circuits," *IEEE Trans. Circuits Syst.*, vol. CAS-30, pp. 762–765, Aug. 1983.

[4] G. Fedi, R. Giomi, A. Luchetta, S. Manetti, and M. C. Piccirilli, "On the application of symbolic techniques to the multiple fault location in low testability analog circuits," *IEEE IEEE Trans. Circuits Syst. II*, vol. 45, pp. 1383–1388, Oct. 1998.

[5] R. Spina and S. Upadhyaya, "Linear circuit fault diagnosis using neuromorphic analyzers," *IEEE Trans. Circuits Syst. II*, vol. 44, pp. 188–196, Mar. 1997.

[6] M. Catelani and M. Gori, "On the application of neural network to fault diagnosis of electronic analog circuits," *Measurement*, vol. 17, pp. 73–80, 1996.

[7] W. Hochwald and J. D. Bastian, "A DC approach for analog fault dictionary determination," *IEEE Trans. Circuits Syst.*, vol. CAS-26, pp. 523–529, July 1979.

[8] K. C. Varghese, J. H. Williams, and D. R. Towill, "Simplified ATPG and analog fault location via a clustering and separability technique," *IEEE Trans. Circuits Syst.*, vol. CAS-26, pp. 496–505, July 1979.

[9] A. McKeon and A. Wakeling, "Fault diagnosis in analogue circuit using AI technique," in *IEEE Int. Test Conf.*, 1989, pp. 118–123.

[10] D. Ying and H. Yigang, "On the application of artiflcial neural networks to fault diagnosis in analog circuits with tolerance," in *5th Int. Conf. Signal Process. Proc., 2000. WCCC-ICSP*, vol. 3, 2000, pp. 1639–1642.

[11] Y. Deng, Y. He, and Y. Sun, "Fault diagnosis of analog circuits with tolerances using artificial neural networks," in *The 2000 IEEE Asia-Pacific Conf. Circuits Syst., 2000*, 2000, pp. 292–295.

[12] J. W. Bandler and A. E. Salama, "Fault diagnosis of analog circuits," *Proc. IEEE*, vol. 73, pp. 1279–1325, Aug. 1985.

[13] M. Catelani and A. Fort, "Fault diagnosis of electronic analog circuits using a radial basis function network classifier," *Measurement*, vol. 28, pp. 147–154, 2000.

[14] L. I. Kuncheva, "On the equivalence between fuzzy and statistical classifiers," *Int. J. Uncertainty, Fuzziness, Knowledge-Based Syst.*, vol. 43, pp. 245–253, 1996.

[15] B. Carse, T. C. Fogarty, and A. Munro, "Evolving fuzzy rules based controllers using genetic algorithms," *Fuzzy Set Syst.*, vol. 80, pp. 273–293, 1996.

[16] O. Cordón, M. J. del Jesus, and F. Herrera, "A proposal on reasoning methods in fuzzy rule-based classification systems," *Int. J. Approx. Reas.*, vol. 20, pp. 21–45, 1999.

[17] F. Klawon and P. E. Klement, "Mathematical analysis of fuzzy classifiers," *Lecture Notes Comput. Sci.*, vol. 1280, pp. 359–370, 1997.

[18] S. Abe and M.-S. Lan, "A method for fuzzy rules extraction directly from numerical data and its application to pattern classification," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 18–28, Feb. 1995.

[19] S. Abe and R. Thawonmas, "A fuzzy classifier with ellipsoidal regions," *IEEE Trans. Fuzzy Syst.*, vol. 5, pp. 358–368, Aug. 1997.

[20] J. C. Bezdek and S. K. Pal, *Fuzzy Models for Pattern Recognition*. Piscataway, NJ: IEEE Press, 1991.

[21] J. A. Leonard, M. A. Kramer, and L. H. Ungar, "Using radial basis function to approximate a function end its error bounds," *IEEE Trans. Neural Networks*, vol. 3, pp. 624–627, July 1992.

[22] D. S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks," *Complex Syst.*, vol. 2, pp. 321–323, 1988.

[23] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Networks*, vol. 2, pp. 302–309, Mar. 1991.

[24] J. Moody and J. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Comput.*, vol. 1, pp. 281–294, 1989.

[25] M. Sugeno, *Industrial Applications of Fuzzy Control*. New York: Elsevier, 1985.

[26] B. Fritzke, "Incremental neuro-fuzzy systems," in *Proc. Applicat. Soft Comput., SPIE Int. Symp. Opt. Sci., Eng. Instrum.*, San Diego, CA, 1997.

[27] ——, "Growing cell structures—A self-organizing network for unsupervised and supervised learning," *Neural Networks*, vol. 7, pp. 1441–1460, 1994.

[28] ——, "The LBG-U method for vector quantization—An improvement over LBG inspired from neural networks," *Neural Process. Lett.*, vol. 5, no. 1, 1997.

[29] T. Poggio and F. Girosi, "A theory of networks for approximation and learning," *Proc. IEEE*, vol. 78, pp. 1481–1497, Sept. 1990.

[30] M. J. D. Powell, "Radial basis functions for multivariable interpolation: A review," in *Algorithms for Approximation*. Oxford, U.K.: Clarendon, 1987, pp. 143–167.

[31] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd ed. New York: Academic, 1990.

Marcantonio Catelani received the degree in electronic engineering from the University of Florence, Italy.

Since 1984, he has been with the Department of Electronic Engineering, now called the Electronic and Telecommunication Department, University of Florence, where he is Associate Professor of reliability and quality control. His current interests are in reliability test and analysis for electronic systems and devices, fault diagnosis, quality control, and instrumentation and measurement, in which his publications are focused.

Ada Fort (M'91) received the Laurea degree in electronic engineering from the University of Florence, Italy, in 1989. She received the Ph.D. degree in nondestructive testing from the same University in 1992.

Since 1997, she has been Assistant Professor at the Department of Information Engineering, University of Siena, Italy. Her current interests concern the development of measurement systems based on chemical sensors and the development of automatic fault diagnosis systems.